E80-10215
JSC-13837 NASA CR-160639

DESIGN SPECIFICATION

FOR

LACIE FORMATTED DOT CARDS IN EOD-LARSYS

Job Order 71-695

TIRF 77-0070

(E80-10215) DESIGN SPECIFICATION FOR LACIE FORMATTED DOT CARDS IN EOD-LARSYS (Lockheed Electronics Co.) 14 p HC A02/MF A01 N80-29793
CSCL 05B G3/43 Unclas 00215

Prepared By

Lockheed Electronics Company, Inc.

Systems and Services Division

Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION

SPACE AND LIFE SCIENCES DIRECTORATE

National Aeronautics and Space Administration

LYNDON B. JOHNSON SPACE CENTER

Houston, Texas

December 1977 LEC-11703

JSC-13837

DESIGN SPECIFICATION
FOR
LACIE FORMATTED DOT CARDS IN EOD-LARSYS

Job Order 71-695

TIRF 77-0070

Prepared by

P. J. Aucoin Jr.

Jeannie Gor

APPROVED BY

LEC

James A Wilkinson

for Philip L. Krumm, Supervisor
Applications Software Section

B L Carroll

B. L. Carroll, Manager
LACIE Development and
Evaluation Department

Prepared by

Lockheed Electronics Company, Inc.

For

Earth Observations Division

Science and Applications Directorate

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

December 1977 LEC-11703

CONTENTS

1. SCOPE

This document contains a preliminary design specification for an augumentation to the Procedure 1 EOD-LARSYS system. This addition involves reading starting and/or bias dots in the LACIE format from the DOTDATA processor.

It is planned to implement this addition on the UNIVAC EXEC II version of EOD-LARSYS and then implement the changes on the Purdue-LARS IBM 370/148 version currently under conversion.

2. APPLICABLE DOCUMENTS

- As-Built Design Specification for EOD-LARSYS Procedure 1
- TIRF: 77-0070

3. SYSTEM DESCRIPTION

3.1 HARDWARE DESCRIPTION

N/A

3.2 SOFTWARE DESCRIPTION

The DOTDATA processor of the EOD-LARSYS system will be expanded to accept dot (field) cards in the LACIE Procedure 1 format. These cards have the general form:

DOT (TYPE) (CATEGORY) {(LACIE NUMBERS)}

where

DOT:	starts in col. 1
TYPE:	= 1 or 2
CATEGORY:	1 character category identifier
LACIE NUMBER:	integer value from 1 to 209.

This addition will require an additional option read by subprogram SET13 and flagged by an additional variable in the DOTVEC labeled common block. A new routine, FLDLAC, will be added to read and decode the dot cards.

In order to specify dots not covered by LACIE numbers, a special code will be implemented. This code will consist of line and sample incrementors added to the LACIE number.

3.2.1 SOFTWARE COMPONENT NO. 1 (SET13)

Subprogram SET13 reads the control cards needed for dot processing.

3.2.1.1 Linkages

SET13 is called by routine DOTDAT and in turn calls utilities NUMBER, NXTCHR, FIND, and ORDER.

3.2.1.2 Interfaces

SET13 interfaces with other routines by means of common blocks DOTVEC, INFORM, and GLOBAL.

3.2.1.3 Inputs

- New/Revised Control Cards

OPTION	LACIE	Turn on LACIE dot format option

3.2.1.4 Outputs

N/A

3.2.1.5 Storage Requirements

TBD

3.2.1.6 Description

The option LACIE will be added to the OPTION control card. The variable named LACIE will be added to the DOTVEC labeled common block. LACIE will be initialized to the value 0. Upon encountering an OPTION LACIE control card, LACIE will be reset to the value 1, indicating LACIE type dot (field) cards will follow the *END* card.

3.2.1.7 Flow Chart

N/A

3.2.1.8 Listing

TBP

3.2.2 SOFTWARE COMPONENT NO. 2 (FLDLAC)

The new subprogram FLDLAC will read and decode the LACIE formatted field (dot) cards.

3.2.2.1 Linkages

FLDLAC will be called by subprogram DOTS if LACIE ≠ 0. Each call to FLDLAC will provide, upon return

1. a dot (field) description (first return)
2. transfer to dot file writing (second return)
3. transfer to dot file writing (third return)

3.2.2.2 Interfaces

FLDLAC will interface with other routines thorugh a calling sequence and common blocks DOTVEC and INFORM.

3.2.2.3 Inputs

Calling Sequence:

SUBROUTINE FLDLAC (FIELDS, STAMNT, $, $, $, IPT, VERTEX)

Parameter	Dimension	In/out	Description
FIELDS	(4,250)	Out	Category name and dot type for dot I stored in FIELDS (1,I) and FIELDS (4,I).
STAMNT	1	In/out	Initially set equal to 1, switch to indicate dots being taken from currently read card.
$			returns to DOTS
$			
$			
IPT	1	In/out	Initially set equal to 1, index number for field vertex information.

Parameter	Dimension	In/out	Description
VERTEX	1000	Out	Vertex information for each dot.

In addition, FLDLAC stores the FLDINF vector in common block DOTVEC with rectangular cc ordinates of field enclosing each dot field.

3.2.2.4 Outputs

N/A

3.2.2.5 Storage Requirements

TBD

3.2.2.6 Description

The DOTDATA processor will be modified to permit reading and processing of dot cards of the form

DOT (TYPE) (CATEGORY NAME) ({LACIE NUMBERS}), ie,

DOT 1 W 2 5 10 29 32 54 110

The present processing uses input training field formats. "TYPE" cards are used to prefix a set of dots. This will remain as the default option. The association between LACIE numbers and training field co ordinates is as follows.

SAMPLE NUMBER

LINE NUMBER	10	20	30 · · ·	190
10	1	2	3	19
20	20			38
30	39			57
:			LACIE NUMBERS	
:				
110	191			209

It is intended that two expansions of the LACIE card format be incorporated. These are

1. free-field locations of all information cards, cols 1-72, data items separated by at least one blank, with the restriction that DOT identifiers start in col 1, and the dot type appear in column 5.

2. In order to cover pixels not included in the LACIE numbering scheme, input dot numbers will be represented as the numerical equivalents of

   $N = L*10^8 + S*10^4 + LACIE$

   where

   LI = #lines to be incremented (up or down) from the line number mapped from the LACIE number. The convention will will be

LI	negative	to	increment up
LI	positive	to	increment down
LI	zero	to	avoid incrementation

   SI = #samples to be incremented (right or left)

SI	negative	to	increment left
SI	positive	to	increment right
SI	zero	to	avoid incrementation.

For example, LI=2, SI= -3, LACIE = 38 yields
$N = 2*10^8 - 3*10^4 + 38 = 199970038$
would correspond to the pixel at (187,22), ie, the pixel at sample number 187 and line number 22.
Letting LI= SI=0, LACIE = 38, obtain
N = 38, correspond to the pixel at (190,20).

Reduction of the value of N to sample and line co ordinates will proceed as follows.

$N_1 = |N|/10^8$ (truncated to integer)

if $|N|-N_1*10^8 \geq 10^7$ set $N_1 = N_1+1$

$LI = N_1 *$ sign (N)

$|N_2| = N-L*10^8$

$N_3 = |N_2|/10^4$ (truncated to integer)

if $|N_2| - N_3*10^4 \geq 10^3$ set $N_3 = N_3+1$

$SI = N_3 *$ sign (N_2)

$LACIE = N_2 - S*10^4$

LR = LACIE ROW# = (LACIE/19 + 1) *10

LS = LACIE COL# = (LACIE - ((LR-1)*19)/10)*10

where truncated divides are specified.

Finally,

L = LR + LI line number corresponding to N

S = LS + SI sample (column) number corresponding to N.

In the scheme to follow, each dot will be considered to be a field. All type 1 dots will occur prior to type 2 dots, ie, the input cards cannot be scrambled with respect to dot type. Otherwise, arbitrary order to cards and LACIE numbers on each card are permitted.

At present, subprogram FLDTYP, called by DOTS, processes dot cards. This routine is not easily modified to accept the LACIE format. Consequently a new subprogram, FLDLAC, will be written and called instead of FLDTYP from DOTS if LACIE $\neq 0$. It will be called from DOTS as

CALL FLDLAC (FIELDS, STAMNT, $100, $510, $520, IPT, VERTEX)

Initialization, at the start of the DOTS routine, will invoke

```
   IPT = 1
STAMNT = 1
NOFLD2 = 0
  TYPE = 1
```

(The description assumes UNIVAC Fortran V conventions, these will be modified tc correspond to IBM Fortran IV conventions when these modifications are added to the converted EOD-LARSYS system). Subprogram FLDLAC will have the following structure.

```
SUBROUTINE FLDLAC (Fields, STAMNT, *, *, *, IPT, VERTEX)
IMPLICIT INTEGER (A-Z)
DIMENSION FIELDS(4,1), VERTEX(1), CARD(62), LDϕTS(30)
LϕGICAL SWITCH
DATA   SWITCH/.TRUE./,   SWCHG/0/,ENDBCD/$END/
INCLUDE   CϕMBK1              ( / INFϕRM/)
INCLUDE   CMBK14              ( / DϕTVEC/)
```

The function of the various parameters is as follows.

IPT — index number for dot (field) vertex information

NOFLD2 — number of fields (dots) for dots of current type (common block INFORM)

SWCHG — number of times dot type has changed. This must be no greater than 1 or an input error will have occurred.

SWITCH — flags a dot type change. The second return will be taken for subsequent writing of a dot field. (internal)

STAMNT — if = 1, a new dot card has been ready if = 2, dots are being processed from a previously read card.

TYPE — dot type being processed (common block DOTVEC)

The calling sequence of FLDLAC is the same as that for FLDTYP, and the meaning of FIELDS and VERTEX remains the same.

```
      IF  (STAMNT.EQ.2)  GΦ  TΦ  30
      IF  (.NOT.SWITCH)  GΦ  TΦ  20

10    READ A CARD, extract TYPES from column 5
      If (TYPE.EQ.TYPES)  GΦ  TΦ  20
      If (SWCHG.NE.0)   error exit
      TYPE=TYPES

20    RE-READ CARD, extract
           CATNM      category name
           NDCARD     #dots on this card
           NDΦTS(I), I=1,NDCARD      dots on this card
      If (NDCARD.EQ.0)  GΦ TΦ 10
      ICNT = 0
      STAMNT = 2
      SWITCH = .TRUE.
      GΦ TΦ 100

30    If (ICNT.LT.NDCARD)     GO TO 100

      STAMNT = 1
      ICNT   = 0

      READ A CARD, extract first 4 characters and store in IDUM,
      extract TYPES

      IF (IDUM,EQ.ENDBCD) RETURN 5

      IF (TYPE.EQ.TYPES)  GΦ TΦ 20

      SWITCH. = .FALSE.
      SWCHG = SWCHG+1
      IF (SWCHG.GT.1)  error exit
      NΦFLD2 = 0; TYPE = TYPES
      IPT = 1
      RETURN 4
```

```
100   INCNT = ICNT+1
      NØFLD2 = NØFLD2+1
      find sample and line numbers S and L from NDØTS (ICNT) as
      described previously.

      Store
      FIELDS (1,NØFLD2) = CATNM
      FIELDS (4,NØFLD2) = 2
      FLDINF (1) = L
      FLDINF (2) = L
      FLDINF (3) = 1
      FLDINF (4) = S
      FLDINF (5) = S
      FLDINF (6) = 1
```

} rectangular bodering field (dot)

```
      VERTEX (IPT)   = S
      VERTEX (IPT+1) = L
      VERTEX (IPT+2) = S
      VERTEX (IPT+3) = L
      IPT = IPT +4
      RETURN
      END
```

Regarding the extraction of dot numbers NDOTS(I), I=1, NDCARD, a new routine, NUMBR, similar to existing function NUMBER will be provided. The differences will be

1. NUMBR will recognize blanks instead of commas as delimiters
2. NUMBR will process only one card of information

3.2.2.7 Flow Chart

TBP

3.2.2.8 Listing

TBP